```
RRRRRRRRRRR      MMM          MMM     SSSSSSSSSSS
RRRRRRRRRRRR     MMM          MMM     SSSSSSSSSSSS
RRRRRRRRRRRR     MMM          MMM     SSSSSSSSSSSS
RRR         RRR  MMMMMM    MMMMMM   SSS
RRR         RRR  MMMMMM    MMMMMM   SSS
RRR         RRR  MMMMMM    MMMMMM   SSS
RRR         RRR  MMM  MMM  MMM  MMM  SSS
RRR         RRR  MMM  MMM  MMM  MMM  SSS
RRR         RRR  MMM  MMM  MMM  MMM  SSS
RRRRRRRRRRRR     MMM          MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM          MMM     SSSSSSSSS
RRRRRRRRRRRR     MMM          MMM     SSSSSSSSS
RRR   RRR        MMM          MMM             SSS
RRR   RRR        MMM          MMM             SSS
RRR   RRR        MMM          MMM             SSS
RRR      RRR     MMM          MMM             SSS
RRR      RRR     MMM          MMM             SSS
RRR      RRR     MMM          MMM             SSS
RRR         RRR  MMM          MMM  SSSSSSSSSSS
RRR         RRR  MMM          MMM  SSSSSSSSSSS
RRR         RRR  MMM          MMM  SSSSSSSSSSS
```

FILEID**RMSFWADEF

```
RRRRRRRR    MM       MM    SSSSSSSS  FFFFFFFFFF  WW      WW    AAAAAA    DDDDDDDD    EEEEEEEEEE  FFFFFFFFFF
RRRRRRRR    MM       MM    SSSSSSSS  FFFFFFFFFF  WW      WW    AAAAAA    DDDDDDDD    EEEEEEEEEE  FFFFFFFFFF
RR      RR  MMMM   MMMM  SS          FF          WW      WW  AA      AA  DD      DD  EE          FF
RR      RR  MMMM   MMMM  SS          FF          WW      WW  AA      AA  DD      DD  EE          FF
RR      RR  MM MM MM MM  SS          FF          WW      WW  AA      AA  DD      DD  EE          FF
RR      RR  MM  MMM  MM  SS          FF          WW      WW  AA      AA  DD      DD  EE          FF
RRRRRRRR    MM       MM    SSSSSS    FFFFFFFF    WW      WW  AA      AA  DD      DD  EEEEEEEE    FFFFFFFF
RRRRRRRR    MM       MM    SSSSSS    FFFFFFFF    WW      WW  AA      AA  DD      DD  EEEEEEEE    FFFFFFFF
RR  RR      MM       MM          SS  FF          WW  WW  WW  AAAAAAAAAA  DD      DD  EE          FF
RR  RR      MM       MM          SS  FF          WW  WW  WW  AAAAAAAAAA  DD      DD  EE          FF
RR    RR    MM       MM          SS  FF          WWWW  WWWW  AA      AA  DD      DD  EE          FF          ....
RR    RR    MM       MM          SS  FF          WWWW  WWWW  AA      AA  DD      DD  EE          FF          ....
RR      RR  MM       MM  SSSSSSSS    FF          WW      WW  AA      AA  DDDDDDDD    EEEEEEEEEE  FF          ....
RR      RR  MM       MM  SSSSSSSS    FF          WW      WW  AA      AA  DDDDDDDD    EEEEEEEEEE  FF          ....

  SSSSSSSS  DDDDDDDD    LL
  SSSSSSSS  DDDDDDDD    LL
SS          DD      DD  LL
SS          DD      DD  LL
SS          DD      DD  LL
SS          DD      DD  LL
  SSSSSS    DD      DD  LL
  SSSSSS    DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
        SS  DD      DD  LL
SSSSSSSS    DDDDDDDD    LLLLLLLLLL
SSSSSSSS    DDDDDDDD    LLLLLLLLLL
```

```
{       $begin  rmsfwadef,V04-000
{
{*****************************************************************************
{*                                                                           *
{*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
{*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
{*  ALL RIGHTS RESERVED.                                                     *
{*                                                                           *
{*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
{*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE    *
{*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
{*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
{*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
{*  TRANSFERRED.                                                             *
{*                                                                           *
{*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
{*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
{*  CORPORATION.                                                             *
{*                                                                           *
{*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
{*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
{*                                                                           *
{*****************************************************************************
{
```

```
;
;       internal rms FWA structure definitions
;
; Modified By:
;
;       V03-032 DGB0077         Donald G. Blair         13-Aug-1984
;               Add FWA$L_UCBSTS.
;
;       V03-031 RAS0321         Ron Schaefer            9-Jul-1984
;               Eliminate access mode itemlist entry for $TRNLNM.
;
;       V03-030 JEJ0046         J E Johnson             05-Jul-1984
;               Increase the length of the quoted buffer to handle a
;               long resultant file specification.
;
;       V03-029 RAS0296         Ron Schaefer            18-Apr-1984
;               Make $PARSE followed by $CREATE on a spooled device work.
;               Add a $GETDVI field for the secondary device name
;               that will be returned in NAM$T_DVI for spooled devices.
;               Make device buffers be only 16 chars for this release.
;
;       V03-028 CDS0002         Christian D. Saether    11-Apr-1984
;               Modify FIBLEN constant to reflect change in fib length.
;
;       V03-027 JWT0166         Jim Teague              12-Mar-1984
;               Shorten the FWA by 124 bytes by getting rid of
;               FWA$T_ATR_LIST.  Add a longword FWA$L_ATR_WORK which
;               will be a pointer to a dynamically allocated work
;               area for ACP attributes.  If FWA$L_ATR_WORK is 0,
;               then a work area is not currently allocated.
;
;       V03-026 RAS0261         Ron Schaefer            6-Mar-1984
;               Delete FWA$V_CUR_VER and rename FWA$V_NULL_NODE to
;               FWA$V_EXP_NODE.
;               Delete separate QUOTED buffer and make NAME buffer
;               big enough for both.  Make FWA$Q_NAME and FWA$Q_QUOTED equal.
;               Grow the FIB buffer.
;
;       V03-025 RAS0234         Ron Schaefer            11-Jan-1984
;               Add definition of $SLBHDEF, revise FWA searchlist fields
;               and filename storage to support multi-input sticky
;               searchlists.  Make device buffers be 256 characters long.
;
;       V03-024 RAS0231         Ron Schaefer            9-Jan-1984
;               Add FWA flag definition for syntax-only parse.
;               This saves probing the NAM block in various places.
;
;       V03-023 RAS0226         Ron Schaefer            29-Dec-1983
;               Add FWA $TRNLNM mode processing back in for correct PPF
;               processing.   This was deleted by RAS0219 incorrectly.
;
;       V03-022 RAS0219         Ron Schaefer            8-Dec-1983
;               Reorganize FWA fields and change BID values:
;               Change SCB$ to FSCB$.
;               Add FWA$T_CDIRxBUF fields for rooted directory names.
;               Add FWA$T_CDEVICEBUF for concealed device name.
```

```
{               Delete FWA$x_XLTBUFy buffers, now dynamically allocated.
{               Delete FWA$T_SWB field, now dynamically allocated.
{               Delete FWA$T_xN fields, now dynamically allocated.
{               Delete FWA$T_CONCEAL_BUF.
{
{       V03-021 RAS0212         Ron Schaefer            15-Nov-1983
{               Add FWA$L_DEV_CLASS for use by RMOPRFLNM/STAPRFLNM
{               in stand-alone BACKUP.
{
{       V03-020 SHZ0001         Stephen H. Zalewski     12-Sep-1983
{               Change the size of SHRFILBUF to 32 bytes, and add an
{               additional buffer of 32 bytes to contain the name of a unique
{               lock name for a file.
{
{       V03-019 KBT0580         Keith B. Thompson       11-Aug-1983
{               Make name and type buffers 1 byte bigger so that the
{               '.' and '{' will not overwrite the next fields
{               Also clean up some constants.
{               Also shorten SLB cause we don't have to save the logical
{               name in the SLB anymore.
{
{       V03-018 KBT0556         Keith B. Thompson       13-Jul-1983
{               Add some fields to scb
{
{       V03-017 KBT0549         Keith B. Thompson       23-Jun-1983
{               Change the ln_flg and sl_flg to ln_flgs and sl_flgs
{
{       V03-016 KBT0536         Keith B. Thompson       1-Jun-1983
{               Add SWB definitions and move some fields around
{
{       V03-015 KBT0528         Keith B. Thompson       25-May-1983
{               Change ITMLIST to ITMLST and fix the item offsets and
{               add the file name buffers
{
{       V03-014 KBT0504         Keith B. Thompson       3-May-1983
{               Add SLB and some related FWA fields. (for search list)
{
{       V03-013 JWH0210         Jeffrey W. Horn         12-Apr-1983
{               Add IDACE, the journal id ACE.
{
{       V03-012 RAS0146         Ron Schaefer            18-Apr-1983
{               Fix FWA$T_FIBBUF to be slightly more tolerant
{               of changes in the ACP FIB length.
{
{       V03-011 KBT0501         Keith B. Thompson       7-Mar-1983
{               Add SCB block and FWA$L_IMPURE_AREA
{
{       V03-010 JWH0190         Jeffrey W. Horn         15-Feb-1983
{               Add VOLNAM field.
{
{       V03-009 KBT0495         Keith B. Thompson       15-Feb-1983
{               Add null_node flag
{
{       V03-008 KBT0484         Keith B. Thompson       1-Feb-1983
{               Make wild buffer long enough for long directory name and
{               add swb pointer
```

```
{
{       V03-007 KBT0456         Keith B. Thompson       7-Jan-1983
{               Put BID, BLN and FWA_PTR fields and make it longword aligned
{
{       V03-006 CDS0001         Christian D. Saether    7-Jan-1983
{               Change FWA$T_FIBBUF and FWA$C_FIBLEN to 56 (new FIB length)
{
{       V03-005 KBT0449         Keith B. Thompson       5-Jan-1983
{               Make user char. a real longword field
{
{       V03-004 JWH0150         Jeffrey W. Horn         13-Dec-1982
{               Modify Journaling work area to implement seperate
{               ACEs for each journal name.
{
{       V03-003 RAS0107         Ron Schaefer            13-Dec-1982
{               Make room for both words of the user file characteristics
{               longword (FWA$W_UCHAR).
{
{       V03-002 KBT0429         Keith B. Thompson       3-Dec-1982
{               Add fwa$q_shrfil descriptor
{
{       V03-001 KBT0400         Keith B. Thompson       8-Nov-1982
{               Unfold all overlaid definitions and add/delete some defs.
{
```

```
{
{       fwa - field definitions
{
{       file work area definitions (fwa)
{
{  the file work area is used for expanding the file
{  name string and setting up the various parameter
{  blocks for interfacing with filacp
{

module $FWADEF;

/*++
/*
/* Flags
/*
/*--


aggregate FWADEF structure fill prefix FWA$;
    FLAGS_OVERLAY union fill;
        FLAGS quadword unsigned;                    /* various parse status flags
        FLD_FLGS structure fill;
            PASSFLGS byte unsigned;                 /* flags for pass only
            FLDFLGS byte unsigned;                  /* flags for fields seen
            WILDFLGS byte unsigned;                 /* flags for wild cards
            PARSEFLGS byte unsigned;                /* flags for parse results
            DIRFLGS byte unsigned;                  /* flags primarily for directory spec
            DIRWCFLGS byte unsigned;                /* directory wild flags
            LNFLGS byte unsigned;                   /* logical name flag byte
            SLFLGS byte unsigned;                   /* search list + rooted directory flags
        end FLD_FLGS;

/*
/* flags for pass
/*

        FLAGS_FIELDS structure fill;
            PASSFLGS_OVERLAY union fill;
                PASSFLGS_BITS structure fill;
                    DUPOR bitfield mask;            /* discard duplicate element
                    NOCOPY bitfield mask;           /* do not copy this field
                    SL_PASS bitfield mask;          /* search list pass
                    RLF_PASS bitfield mask;         /* set if applying related file defaults
                    FNA_PASS bitfield mask;         /* set if primary name string parse pass
                    NAM_DVI bitfield mask;          /* set if open by name block
                    EXP_NODE bitfield mask;         /* explicit node has been seen, null or normal
                end PASSFLGS_BITS;

/*
/* flags for fields seen
/*

                FLAGS_BITS0 structure fill;
                    FILL_1 bitfield length 8 fill prefix FWADEF tag $$;/* start at byte 1
                    FILL_2 bitfield length 3 fill prefix FWADEF tag $$;/* spare
```

```
                VERSION bitfield mask;              /* set if version seen
                TYPE bitfield mask;                 /* set if type seen
                NAME bitfield mask;                 /* set if name seen
                DIR bitfield mask;                  /* set if directory spec seen
                DEVICE bitfield mask;               /* set if device seen
            end FLAGS_BITS0;

/*
/* flags for wild cards
/*

            FLAGS_BITS1 structure fill;
                FILL_3 bitfield length 16 fill prefix FWADEF tag $$;/* start at byte 2
                EXP_VER bitfield mask;              /* set if explicit version
                EXP_TYPE bitfield mask;             /* set if explicit type
                EXP_NAME bitfield mask;             /* set if explicit name
                WC_VER bitfield mask;               /* set if wildcard (*) version
                WC_TYPE bitfield mask;              /* " type
                WC_NAME bitfield mask;              /* " name
                EXP_DIR bitfield mask;              /* set if explicit directory
                EXP_DEV bitfield mask;              /* set if explicit device
            end FLAGS_BITS1;

/*
/* flags for parse results
/*

            FLAGS_BITS2 structure fill;
                FILL_4 bitfield length 24 fill prefix FWADEF tag $$;/* start at byte 3
                WILDCARD bitfield mask;             /* set if any wildcard seen
                NODE bitfield mask;                 /* set if node name seen
                QUOTED bitfield mask;               /* set is quoted string seen
                                                    /* (valid only if node set and no fldflgs)
                GRPMBR bitfield mask;               /* set if directory in [grp,mbr] format
                WILD_DIR bitfield mask;             /* inclusive or of directory wild cards
                DIR_LVLS bitfield mask length 3;    /* # of directory sublevels (0 = ufd only)
            end FLAGS_BITS2;

/*
/* flags primarily for directory spec
/*

            FLAGS_BITS3 structure fill;
                FILL_5 bitfield length 32 fill prefix FWADEF tag $$;/* start at byte 4
                DIR1 bitfield;                      /* ufd level directory or group seen
                DIR2 bitfield;                      /* sfd level 1 directory or member seen
                FILL_6 bitfield length 6 fill prefix FWADEF tag $$;/* additional sub directory level flags
            end FLAGS_BITS3;

/*
/* directory wild flags
/*

            FLAGS_BITS4 structure fill;
                FILL_7 byte dimension 5 fill prefix FWADEF tag $$;/* start at byte 5
                WILD_UFD bitfield;                  /* the dir1 spec was a wild card
```

```
                WILD_SFD1 bitfield;                     /* the dir2 spec was a wild card
                FILL_8 bitfield length 6 fill prefix FWADEF tag $$;/* additional sub directory wildcard flags
            end FLAGS_BITS4;
            FLAGS_BITS5 structure fill;
                FILL_9 byte dimension 5 fill prefix FWADEF tag $$;/* alternate definition for dir1 and dir2
                WILD_GRP bitfield;                      /* the grp spec contained a wild card
                WILD_MBR bitfield;                      /* the mbr spec contained a wild card
            end FLAGS_BITS5;

/*
/* logical name flag and miscellaneous byte
/*

            FLAGS_BITS6 structure fill;
                FILL_10 byte dimension 6 fill prefix FWADEF tag $$;/* start at byte 6
                LOGNAME bitfield;                       /* a logical name has been seen this pass
                                                        /*  (note: this byte is saved as context
                                                        /*  when processing [.dir-list] format)
                OBJTYPE bitfield;                       /* set if quoted string is of the
                                                        /*  "objecttype=..." form
                                                        /*  (valid only if quoted set)
                NETSTR bitfield;                        /* set if quoted string is of the
                                                        /*  "objecttype=taskname/..." form
                                                        /*  (valid only if quoted and objtype set)
                DEV_UNDER bitfield;                     /* device name was prefixed with an underscore
                FILEFOUND bitfield;                     /* true if at least one file found by search
                REMRESULT bitfield;                     /* use resultant string returned by fal
                SYNTAX_CHK bitfield;                    /* syntax-only checking is requested (NAM$V_SYNCHK set)
            end FLAGS_BITS6;

/*
/* search list and rooted directory flag byte
/*

            FLAGS_BITS7 structure fill;
                FILL_11 byte dimension 7 fill prefix FWADEF tag $$;/* start at byte 7
                SLPRESENT bitfield;                     /* search list present
                CONCEAL_DEV bitfield;                   /* concealed device present
                ROOT_DIR bitfield;                      /* root directory present
                DFLT_MFD bitfield;                      /* default MFD string inserted, due to [-]
                EXP_ROOT bitfield;                      /* explicit root directory
            end FLAGS_BITS7;

        end PASSFLGS_OVERLAY;
    end FLAGS_FIELDS;
end FLAGS_OVERLAY;

/*
/* Value for all filename elements except node
/*

        constant ALL        equals
            ( ((FWA$M_DEVICE!
            FWA$M_DIR!
            FWA$M_NAME!
            FWA$M_TYPE!
```

```
            FWA$M_VERSION)@-8)  )
          prefix FWA tag $C;

/*
/* constant for all flags that vary per parsing pass
/*

        constant ALLPASS    equals
            ( FWA$M_DUPOK!
              FWA$M_FNA_PASS!
              FWA$M_RLF_PASS )
         prefix FWA tag $C;

/*++
/*
/* Misc. Fields
/*
/*--

    BID byte unsigned;                                  /* bid
    constant BID        equals 40  prefix FWA tag $C;   /* bid of fwa
    BLN byte unsigned;                                  /* bln
    DIRTERM byte unsigned;                              /* directory spec terminator (']' or '>')
    ROOTERM byte unsigned;                              /* root directory spec terminator (']' or '>')
    ESCSTRING_OVERLAY union fill;
        ESCSTRING longword unsigned;                    /* escape equivalence string
        ESCSTRING_FIELDS structure fill;
            ESCFLG byte unsigned;                       /* set to the char <esc> if an escape string
                                                        /* seen, zero otherwise
            ESCTYP byte unsigned;                       /* escape 'type' byte
            ESCIFI word unsigned;                       /* escape ifi value
        end ESCSTRING_FIELDS;
    end ESCSTRING_OVERLAY;
    FIB quadword unsigned;                              /* fib descriptor
    DEVBUFSIZ longword unsigned;                        /* device buffer size
    DEV_CLASS longword unsigned;                        /* device class
    RECSIZ longword unsigned;                           /* blocked record size
    UNIT longword unsigned;                             /* device unit number
    UIC longword unsigned;                              /* file owner uic
    PRO word unsigned;                                  /* file protection word
    DIRLEN byte unsigned;                               /* overall directory spec length
    SUBNODCNT byte unsigned;                            /* number of secondary (sub) node specs found
    DIRBDB longword unsigned;                           /* address of directory file bdb
    LOOKUP longword unsigned;                           /* address of new directory cache node
    DEVNODADR longword unsigned;                        /* address of device directory cache node
    DIR quadword unsigned;                              /* directory name scratch buffer
    UCHAR_OVERLAY union fill;
        UCHAR longword unsigned;                        /* user characteristics longword
        UCHAR word unsigned;
    end UCHAR_OVERLAY;
    FWA_PTR longword unsigned;                          /* pointer to second fwa if any ($RENAME)
    SWB_PTR longword unsigned;                          /* pointer to swb
    BUF_PTR longword unsigned;                          /* address of temporary buffer
    IMPURE_AREA longword unsigned;                      /* saved R11 (rm$xpfn only)
    ATR_WORK longword unsigned;                         /* pointer to work area for ACP attributes
                                                        /*  (zero if one not currently allocated)
```

```
/***
/*
/* Logical name and search list fields
/*
/*--

/*
/* Item list block for logical name services
/*

    ITMLST_OVERLAY union fill;
        ITMLST character length 64;                 /* logical name item list
        ITMLST_FIELDS structure fill;
            ITM_INDEX character length 12;          /* index
            ITM_ATTR character length 12;           /* attributes
            ITM_STRING character length 12;         /* string
            ITM_MAX_INDEX character length 12;      /* max index
            ITM_END longword unsigned;              /* terminating longword
        end ITMLST_FIELDS;
    end ITMLST_OVERLAY;

/*
/* Logical name translation fields
/*

    BUFFLG byte unsigned;                           /* flag for which translation buffer is in use
                                                    /* (0 = buf2 in use, -1 = buf1 in use)
    XLTMODE byte unsigned;                          /* mode of translation on input to $TRNLNM
                                                    /* mode of equivalence string on output from $TRNLNM
    XLTSIZ word unsigned;                           /* length of equivalence string
    XLTBUFF1 longword unsigned;                     /* primary translation buffer descriptor
    XLTBUFF2 longword unsigned;                     /* secondary translation buffer descriptor

/*
/* SLBH and SLB pointers
/*

    SLBH_PTR longword unsigned;                     /* current SLB list
    SLB_PTR longword unsigned;                      /* current SLB list

    SLBH_FLINK longword unsigned;                   /* SLBH que fwd link
    SLBH_BLINK longword unsigned;                   /* SLBH que back link

/*
/* Fake SLB -   NOTE:        This MUST be the size of SLB$C_BLN
/*                       The field FWA$B_LEVEL must be at the same offset
/*                       as SLB$Q_LEVEL would be.  (It sounds like a real
/*                       hack but it works very nicely)
/*

    SLB_OVERLAY union fill;
        SLB character length 24;                    /* space for SLB$C_BLN
        SLB_FIELDS structure fill;
            FILL_17 byte dimension 11 fill prefix FWADEF tag $$;
            LEVEL byte unsigned;                    /* recursion level
        end SLB_FIELDS;
```

```
    end SLB_OVERLAY;
/*
/* Logical name descriptor
/*
    LOGNAM quadword unsigned;                   /* logical name descriptor
```

```
/***
/*
/*  descriptors for parsed filename elements
/*
/*
/*        The descriptors are defined as:
/*
/*        ------------------------------------------------
/*        |        flags          |              length  |
/*        |-----------------------------------------------|
/*        |                     address                   |
/*        ------------------------------------------------
/*
/*
/*        The flags are defined by FSCB$V_flag in $FSCBDEF
/*
/***--

    NODE quadword unsigned;                                 /* node name (actually node spec list) descriptor
                                                            /* (the associated buffer is fwa$t_nodebuf)
    constant MAXNODNAM  equals 6  prefix FWA tag $C;        /* max node name size
    constant MAXLNDNAM  equals 15  prefix FWA tag $C;       /* max logical node name size
    constant MAXNODLST  equals 127  prefix FWA tag $C;      /* max node spec list size (concatenated node specs)

/*
/*        device name descriptor
/*

    DEVICE quadword unsigned;                               /* device name descriptor
    constant MAXDEVICE  equals 255  prefix FWA tag $C;      /* max device name size

    CONCEAL_DEV quadword unsigned;                          /* concealed device descriptor

/*
/*        directory name descriptors   NOTE:  The two sets of directory
/*                                            descriptors must be contigous
/*                                            or RMS$SETDID will break
/*

    CDIR1 quadword unsigned;                                /* concealed top directory descriptors
    CDIR2 quadword unsigned;                                /* concealed subdirectory 1
    CDIR3 quadword unsigned;                                /* " " 2
    CDIR4 quadword unsigned;                                /* " " 3
    CDIR5 quadword unsigned;                                /* " " 4
    CDIR6 quadword unsigned;                                /* " " 5
    CDIR7 quadword unsigned;                                /* " " 6
    CDIR8 quadword unsigned;                                /* " " 7
    constant MAXCDIR    equals 8  prefix FWA tag $C;        /* max number of concealed directories
    DIR1 quadword unsigned;                                 /* top level directory descriptors
    DIR2 quadword unsigned;                                 /* subdirectory 1
    DIR3 quadword unsigned;                                 /* " 2
    DIR4 quadword unsigned;                                 /* " 3
    DIR5 quadword unsigned;                                 /* " 4
    DIR6 quadword unsigned;                                 /* " 5
    DIR7 quadword unsigned;                                 /* " 6
    DIR8 quadword unsigned;                                 /* " 7
```

```
    constant MAXSUBDIR  equals 7  prefix FWA tag $C;    /* max number of sub directories
    constant MAXDIRLEN  equals 255 prefix FWA tag $C;   /* max size of total directory spec
                                                        /*  should be:  top + subdir   39 * 8
                                                        /*              dots between    7
                                                        /*              delimiters      2
                                                        /*              ----------------
                                                        /*              total         321
/*
/*      The filename, filetype and fileversion descriptors MUST be contiguous
/*
/*        file name descriptor
/*

    NAME_QUOTED union fill;
        NAME quadword unsigned;                         /* file name descriptor
        QUOTED quadword unsigned;                       /* quoted string descriptor
    end NAME_QUOTED;
    constant MAXNAME    equals 39  prefix FWA tag $C;   /* max file name size
    constant MAXQUOTED  equals 255  prefix FWA tag $C;  /* max quoted string size

/*
/*        file type descriptor
/*

    TYPE quadword unsigned;                             /* file type descriptor
    constant MAXTYPE    equals 39  prefix FWA tag $C;   /* max file type size

/*
/*        file version number descriptor
/*

    VERSION quadword unsigned;                          /* file version descriptor
    constant MAXVER     equals 6  prefix FWA tag $C;    /* maximum version

    RNS quadword unsigned;                              /* resultant name string descriptor
    constant MAXRNS     equals 86  prefix FWA tag $C;   /* max resultant name string size

    SHRFIL quadword unsigned;                           /* shared file device descriptor (readable form)

    SHRFIL_LCK quadword unsigned;                       /* shared file device descriptor (unreadable form - used for lock name)

    AS_SHRFIL quadword unsigned;                        /* secondary device descriptor (readable form)

    STATBLK_OVERLAY union fill;
        STATBLK character length 10;                    /*
        STATBLK_FIELDS structure fill;
            SBN longword unsigned;                      /* starting lbn if contiguous
            HBK longword unsigned;                      /* high vbn
            constant STATBLK    equals 10  prefix FWA tag $C;/* define length of statistics block
        end STATBLK_FIELDS;
    end STATBLK_OVERLAY;
    FILL_14 word fill prefix FWADEF tag $$;             /* spare

/*
/*  node descriptors
```

```
/*

    NODE1 quadword unsigned;                            /* primary node spec descriptor
                                                        /* (the associated buffer is fwa$t_nodebuf)
    NODE2 quadword unsigned;                            /* secondary (sub) node spec descriptors (1-7)
    NODE3 quadword unsigned;                            /* note: bytes 2-3 of each of these descriptors
    NODE4 quadword unsigned;                            /* contains the flags word that is output
    NODE5 quadword unsigned;                            /* from nxtfld subroutine in rm0xpfn
    NODE6 quadword unsigned;                            /* note: fwa$q_node1 thru 'fwa$q_node8'
    NODE7 quadword unsigned;                            /* describe the same string as does
    NODE8 quadword unsigned;                            /* fwa$q_node
    constant BLN_FWA equals . prefix FWA$ tag K;        /* length of fwa
    constant BLN_FWA equals . prefix FWA$ tag C;        /* length of fwa
    constant MAXSUBNOD  equals 7  prefix FWA tag $C;    /* max number of secondary (sub) node specs
```

```
/*++
/*
/*  buffers for parsed filename elements
/*
/*--

    FIBBUF character length 76;                         /* fib buffer
    constant FIBLEN     equals 76 prefix FWA tag $C;    /* fib buffer size

    RNM_FID character length 6;                         /* saved fid for rename directory check

/*
/*  directory name buffers
/*
/*  NOTE:  These buffers must be contiguous
/*

    DIR1BUF character length 39;                        /* ufd level (or group)
    DIR2BUF character length 39;                        /* 1st sfd level (or member)
    DIR3BUF character length 39;                        /* subdirectory 2
    DIR4BUF character length 39;                        /* subdirectory 3
    DIR5BUF character length 39;                        /* subdirectory 4
    DIR6BUF character length 39;                        /* subdirectory 5
    DIR7BUF character length 39;                        /* subdirectory 6
    DIR8BUF character length 39;                        /* subdirectory 7
    constant DIRBUFSIZ  equals 39  prefix FWA tag $C;   /* size of each directory buffer

/*
/*  rooted directory name buffers
/*
/*  NOTE:  These buffers must be contiguous
/*

    CDIR1BUF character length 39;                       /* ufd level (or group)
    CDIR2BUF character length 39;                       /* 1st sfd level (or member)
    CDIR3BUF character length 39;                       /* subdirectory 2
    CDIR4BUF character length 39;                       /* subdirectory 3
    CDIR5BUF character length 39;                       /* subdirectory 4
    CDIR6BUF character length 39;                       /* subdirectory 5
    CDIR7BUF character length 39;                       /* subdirectory 6
    CDIR8BUF character length 39;                       /* subdirectory 7

/*
/*  NOTES: 1. The following buffers must be contiguous as eventually the
/*            type and version are appended to the name string
/*
/*         2. The name buffer and the type buffer must be 1 byte larger then
/*            the max name and type size (resp) because xpfn writes the
/*            name and type terminators in the buffer at the end of the string
/*

    NAMEBUF character length 256;                       /* file name/quoted string buffer

    TYPEBUF character length 40;                        /* file type buffer

    VERBUF character length 6;                          /* file version buffer
```

```
    UCBSTS longword unsigned;                   /* ucb$l_sts field for prim device

    UNDER_DEV byte unsigned;                    /* character "_" stored here
    DEVICEBUF character length 255;             /* device name buffer

    CDEVICEBUF character length 256;            /* concealed device name buffer

    UNDER_NOD byte unsigned;                    /* character "_" stored here
    NODEBUF character length 127;               /* node name buffer

    WILD character length 48;                   /* scratch field used by RMOWILD
                                                /*   size =       count      1
                                                /*                name      39
                                                /*                .dir;*     6
                                                /*                spare      2
                                                /*                ----------
                                                /*                          48

    SHRFILBUF character length 16;              /* shared file device id buffer (readable form)

    SHRFIL_LCKNAM character length 16;          /* shared file device id buffer (unreadable form - used for lock name)

    AS_SHRFILBUF character length 16;           /* secondary device id buffer (readable form)

    BIJNL quadword unsigned;                    /* descriptor of BI journal name
    AIJNL quadword unsigned;                    /* descriptor of AI journal name
    ATJNL quadword unsigned;                    /* descriptor of AT journal name

    BIACE_OVERLAY union fill;
        BIACE character length 20;              /* BI journal name ACE
        BIACE_FIELDS structure fill;
            FILL_19 byte dimension 4 fill prefix FWADEF tag $$;
            BIJNLN character length 16;
        end BIACE_FIELDS;
    end BIACE_OVERLAY;

    AIACE_OVERLAY union fill;
        AIACE character length 20;              /* AI journal name ACE
        AIACE_FIELDS structure fill;
            FILL_20 byte dimension 4 fill prefix FWADEF tag $$;
            AIJNLN character length 16;
        end AIACE_FIELDS;
    end AIACE_OVERLAY;

    ATACE_OVERLAY union fill;
        ATACE character length 20;              /* AT journal name ACE
        ATACE_FIELDS structure fill;
            FILL_21 byte dimension 4 fill prefix FWADEF tag $$;
            ATJNLN character length 16;

        end ATACE_FIELDS;
    end ATACE_OVERLAY;

    IDACE_OVERLAY union fill;
        IDACE character length 32;              /* Journal ID ACE
        IDACE_FIELDS structure fill;
```

```
            FILL_22 byte dimension 4 fill prefix FWADEF tag $$;
            JNLID_OVERLAY union fill;
                JNLID character length 28;              /* complete journal ID
                JNLID_FIELDS structure fill;
                    VOLNAM character length 12;         /* volume lable of media file resides on
                    FID character length 6;             /* file-id
                    FILL_23 byte dimension 2 fill prefix FWADEF tag $$;
                    ID_DATE quadword unsigned;          /* id time stamp

                end JNLID_FIELDS;
            end JNLID_OVERLAY;
        end IDACE_FIELDS;
    end IDACE_OVERLAY;

    constant BLN_BUF equals . prefix FWA$ tag K;        /* length of fwa and buffers
    constant BLN_BUF equals . prefix FWA$ tag C;        /* length of fwa and buffers
    constant BLN equals . prefix FWA$ tag K;            /* length of fwa and buffers
    constant BLN equals . prefix FWA$ tag C;            /* length of fwa and buffers

end FWADEF;

end_module $FWADEF;
```

```
module $SLBHDEF;
/*
/*      SLBH        - Search List Header Block
/*


aggregate SLBHDEF structure fill prefix SLBH$;
    FLINK longword unsigned;                            /* forward link
    BLINK longword unsigned;                            /* backward link
    BID byte unsigned;                                  /* block ID
    constant BID        equals 43  prefix SLBH tag $C;  /* ID
    BLN byte unsigned;                                  /* length
    PASSFLGS byte unsigned;                             /* flags for FWA$B_PASSFLGS
    STR_LEN byte unsigned;                              /* string length
    SLB_QUE longword unsigned;                          /* ptr to SLB queue
    NAM_FNB longword unsigned;                          /* saved FNB from RLF file
    STRING character length 0;                          /* start of string
    constant BLN equals . prefix SLBH$ tag K;           /* length of SLBH
    constant BLN equals . prefix SLBH$ tag C;           /* length of SLBH
end SLBHDEF;

end_module $SLBHDEF;
```

```
module $SLBDEF;
/*
/*      SLB       - Search List Block
/*


aggregate SLBDEF structure fill prefix SLB$;
    FLINK longword unsigned;                            /* forward link
    BLINK longword unsigned;                            /* backward link
    BID byte unsigned;                                  /* block ID
    constant BID        equals 41  prefix SLB tag $C;   /* ID
    BLN byte unsigned;                                  /* length
    FLAGS_OVERLAY union fill;
        FLAGS byte unsigned;                            /* flags
        FLAGS_BITS structure fill;
            REALSLB bitfield mask;                      /* "Real" SLB as opposed to the fake FWA one
        end FLAGS_BITS;
    end FLAGS_OVERLAY;
    LEVEL byte unsigned;                                /* recursion level
    INDEX longword unsigned;                            /* translation index
    MAX_INDEX longword unsigned;                        /* max translation index
    ATTR longword unsigned;                             /* attributes flags
    constant BLN equals . prefix SLB$ tag K;            /* length of SLB
    constant BLN equals . prefix SLB$ tag C;            /* length of SLB
end SLBDEF;

end_module $SLBDEF;
```

```
module $FSCBDEF;
/*
/*      FSCB - FileScan control block
/*
/*      This block is passed to PARSE_STRING from XPFN and RMS$FILESCAN
/*


/*
/*      The descriptors are defined as:
/*
/*      ------------------------------------------------
/*      |       flags          |        length         |
/*      |-----------------------------------------------|
/*      |                     address                   |
/*      ------------------------------------------------
/*
/*


/*
/*      descriptor flags
/*
/*      These flags are used through out the RMS file name parsing routines.
/*      The flags can be found in all of the field descriptors.
/*
/*      NOTE: The flag ELIPS must be the first bit in the second word.
/*            It is referenced this way in RMOWILD and other places
/*


aggregate FSCBDEF  union fill prefix FSCB$;
    FSCBDEF_BITS structure fill;
        FIL_1 bitfield length 16 fill prefix FSCBDEF tag $$;/* flags are defined in high word of descriptor
        ELIPS bitfield mask;                            /* elipssis was detected in directory (dir)
        WILD bitfield mask;                             /* a wild card was detected (dir,name,type,ver)
        ACS bitfield mask;                              /* access control string in node name (node)
        QUOTED bitfield mask;                           /* quoted file spec (name)
        NULL bitfield mask;                             /* field was null (terminator only) (all)
        PWD bitfield mask;                              /* password masked out (set in xpfn) (node)
        GRPMBR bitfield mask;                           /* group,member format directory (dir)
        MINUS bitfield mask;                            /* minus directory field (dir)
        CONCEAL bitfield mask;                          /* name was concealed (dev)
        MFD bitfield mask;                              /* MFD directory (set in xpfn) (dir)
        ROOTED bitfield mask;                           /* directory was a root directory (dir)
    end FSCBDEF_BITS;
end FSCBDEF;

/*
/*        FSCB
/*

    aggregate FSCBDEF1 structure fill prefix FSCB$;
    FLDFLAGS_OVERLAY union fill;
        FLDFLAGS byte unsigned;                         /* field flags
        FLDFLAGS_BITS structure fill;
            NODE bitfield mask;
```

```
            DEVICE bitfield mask;
            ROOT bitfield mask;
            DIRECTORY bitfield mask;
            NAME bitfield mask;
            TYPE bitfield mask;
            VERSION bitfield mask;
        end FLDFLAGS_BITS;
    end FLDFLAGS_OVERLAY;
    NODES byte unsigned;                                /* number of nodes in spec
    ROOTS byte unsigned;                                /* number of root directories in spec
    DIRS byte unsigned;                                 /* number of directories in spec
    FILESPEC quadword unsigned;                         /* full file spec
    NODE quadword unsigned;                             /* full node list spec
    DEVICE quadword unsigned;                           /* device spec
    ROOT quadword unsigned;                             /* full root directory list spec
    DIRECTORY quadword unsigned;                        /* full directory list spec
    NAME quadword unsigned;                             /* file name
    TYPE quadword unsigned;                             /* file type
    VERSION quadword unsigned;                          /* file version
    NODE1 quadword unsigned;                            /* the NODEn descriptors must be contiguous
    NODE2 quadword unsigned;
    NODE3 quadword unsigned;
    NODE4 quadword unsigned;
    NODE5 quadword unsigned;
    NODE6 quadword unsigned;
    NODE7 quadword unsigned;
    NODE8 quadword unsigned;
    constant MAXNODE    equals 8  prefix FSCB tag $C;   /* max number of node descriptors
    ROOT1 quadword unsigned;                            /* the ROOTn descriptors must be contiguous
    ROOT2 quadword unsigned;
    ROOT3 quadword unsigned;
    ROOT4 quadword unsigned;
    ROOT5 quadword unsigned;
    ROOT6 quadword unsigned;
    ROOT7 quadword unsigned;
    ROOT8 quadword unsigned;
    constant MAXROOT    equals 8  prefix FSCB tag $C;   /* max number of root descriptors
    DIRECTORY1 quadword unsigned;                       /* the DIRECTORYn descriptors must be contiguous
    DIRECTORY2 quadword unsigned;
    DIRECTORY3 quadword unsigned;
    DIRECTORY4 quadword unsigned;
    DIRECTORY5 quadword unsigned;
    DIRECTORY6 quadword unsigned;
    DIRECTORY7 quadword unsigned;
    DIRECTORY8 quadword unsigned;
    constant BLN equals . prefix FSCB$ tag K;
    constant BLN equals . prefix FSCB$ tag C;
    constant MAXDIR     equals 8  prefix FSCB tag $C;   /* max number of directory descriptors
    end FSCBDEF1;

end_module $FSCBDEF;
```

```
module $SWBDEF;
/*
/* Directory string work buffer for wild card directory processing
/*

aggregate SWBDEF structure fill prefix SWB$;
    FLAGS_OVERLAY union fill;
        FLAGS byte unsigned;                                /* flags (must be first)
        FLAGS_BITS structure fill;
            ELLIPSIS bitfield mask;                         /* ellipsis
            BOUNDED bitfield mask;                          /* ellipsis bounded
            WILD bitfield mask;                             /* wild name
            DELIMITER bitfield mask;                        /* following delimiter
            TRAVERSE bitfield mask;                         /* should skip subtree
            FIRST bitfield mask;                            /* first time through
            ELLIPSIS_EXISTS bitfield mask;                  /* dir spec contains ...
            VALID_DID bitfield mask;                        /* FIB DID is valid
        end FLAGS_BITS;
    end FLAGS_OVERLAY;
    PATLEN byte unsigned;                                   /* length of current token
    PPOS byte unsigned;                                     /* position in pattern
    TOKENS_LEFT byte unsigned;                              /* number of non ... tokens left
    MINIMUM byte unsigned;                                  /* minimum level for success
    MAXIMUM byte unsigned;                                  /* maximum level for success
    FIRST_E byte unsigned;                                  /* token ! of first ellipsis
    DIRWCFLGS byte unsigned;                                /* FWA$B_DIRWCFLGS on entry
    BID byte unsigned;                                      /* block ID
    constant BID        equals 42  prefix SWB tag $C;       /* ID
    BLN byte unsigned;                                      /* length
    FILL_1 word fill prefix SWBDEF tag $$;                  /* spare
    PATTERN quadword unsigned;                              /* descriptor of pattern
    SCRATCH_PAT longword unsigned;                          /* scratch copy of first longword
    SCRATCH_BUF character length 48;                        /* scratch temp buffer (same size as FWA$T_WILD)

    PATTERN_BUF character length 256;                       /* should be: FWA$C_MAXDIRLEN-2,
    constant 'BLN' equals . prefix SWB$ tag K;
    constant 'BLN' equals . prefix SWB$ tag C;
                                                            /*  wild dir spec

end SWBDEF;

end_module $SWBDEF;
```